Please amend the Specification according to the following marked up paragraphs.

[0041] If the MA 30 can download the new file structure 500 (FIG. 9) to accommodate the e-cash, it then does so 152 ~~154~~. According to this embodiment, the MA then operates as a slave to the RCT Bluetooth® air modem, receiving messages and passing relevant ones to the SIM 154 ~~152~~. The data associated with loading money is transmitted over the secure data stream and the money is loaded into the SIM card 156. It is generally desirable to timestamp the transaction and update some sort of log file, then terminate the session 158. It may be possible to also reverse the master-client relationship between the MA 30 and the RCT 20. The file structure 500 may support multiple purses 508.1-N (Fig. 9), and so the e-cash is loaded into the appropriate purse. In this scenario, the MA 30 operates as a slave to the RCT Bluetooth® modem 21.1 (Fig. 4A) which receives the communicated messages and passes them along to the SIM 40.

[0047] Given the complexity of the encryption task, additional hardware is required in the form of cryptography engine 46, and the associated S/W to manage, dispatch, encrypt, and decrypt), and a Java Virtual Machine (JVM)™ 522 (which is a Java™ interpreter that executes the byte codes on a particular platform), including Java™ Script Interpreter (JSI) 524. The JVM™ 522 differs from the use of other programming languages in a microprocessor context in the sense that it enables a virtual machine. In most programming languages, the developer must compile the code to support an executable environment, which doesn't provide for much flexibility if the designer would like to expand the scope and functionality of an embedded system. In a JVM<u>TM</u>, the complier converts the code into something commonly referred to as Java code bytes which are then sent off to the ~~Java Script Interpreter (~~JSI [[)]] 524, which parses and runs the ~~java~~ <u>Java™</u> applets. Other forms of command interpreters could be present as well. A traditional SIM card also includes a portion for IMSI Rights Management 520 and includes some form of Common File Management and File Structure Management 502 for the application layers 504, 506, 508.1-N, 510.1-N that refers to a common file structure needed to ensure that space is effectively managed and that all other applications can peacefully coexist.

[0054] The Fig. 4A block diagram of an exemplary RCT 20 shows the generic embedded microcontroller 22 running an operating system that manages the on board resources, such as: a memory 28 (e.g., flash memory and SRAM), a keyboard (or other input mechanism) 26 (by scanning and encoding user input), ~~and~~ a display 24 (by e.g., updating a screen)<u>, and a power management module 29</u>. The I/O ports 21 are commonly referred to as modem ports that are currently used to handle any communications payload during a typical communications session. The various I/O ports 21.1-21.6 can be better thought of as the physical hardware or modem transceivers that are used primarily to handle Bluetooth®, WiFi®, IRDA, W-CDMA or GSM, USB, or other similar types of communications sessions. The hardware depends on an operating system that controls the environment and manages all of the hardware resources, partitions memory, and controls the data traffic into and out of the device, interrupt processing, and security and key encryption.

[0055] As shown in the block diagram of Fig. 4B of the MA 30 that comprises, e.g., the Bluetooth® transceiver 31.1 (as well as a GSM Transceiver 31.2 and a GPRS Transceiver 31.3) which are tied to the microcontroller 32. The MA 30, just like the RCT 20, has the microcontroller 32 running an operating system that manages the handling of a Bluetooth® communication request and has a memory and power management module 36. The micro-controller 32 establishes the session, and handles the inter-processor communications needed to transfer e-cash to the SIM 40, and complete the session.

[0067] According to this embodiment, the game Virtual Slots is downloaded to the MA 30 from the gaming server 74.2. All events occur in pseudo real-time. The game applet 510.1 is downloaded to the SIM card 40, and runs on top of a ~~Java Virtual Machine (~~JVM$^{\text{TM}}$[[)]] 522 environment on the MA 30 (i.e., the game is no longer talking to the server 74.2). The only time that the game applet 510.1 needs to access the server 74.2 is to record and request a win payout, or to request a different game.